FILIGRANE (FlexIbLe IPR for Software AGent ReliANcE)

A security framework for trading of mobile code in Internet

Mehrdad Jalali [1]
Fraunhofer Institut für Graphische Datenverarbeitung-IGD
Runderturmstr.6
64283 Darmstadt GERMANY

Gael Hachez
UCL / DICE / Crypto Group
Place du Levant, 3
B-1348 Louvain-la-Neuve
Belgium

Christophe Vasserot
Thomson-CSF
Core Technique&Technology Unit
160 Boulevard de Valmy, 92704
Colombes, FRANCE

ABSTRACT

In This paper we give an overview about our security framework developed in the ESPRIT Project FILIGRANE. The goal of this project is to develop an integrated security framework for mobile code commerce. The term mobile code includes all kind of mobile Java software (applets and agents and Java beans, cardlets etc.). Our framework has been developed to be used in the Internet and in particular in the World Wide Web. It is based on Smart Card, Public Key Cryptography and ERMS (Electronic Right Management System). The first integrated version of the FILIGRANE framework now exists and will be available shortly as a web Demonstrator.

Keywords

mobile agent security, electronic commerce, electronic right management, copyright protection, software piracy, reverse engineering

1. Introduction

Reverse engineering, once the secret of experienced hackers, is going to be mainstream with Java. Software piracy has proved to be a very significant issue in the mobile code software business today. In most cases, pirates analyse how existing software works, either to change it or to duplicate it. Any language can be subject to reverse engineering but some features of Java, namely portability and object orientation, have made it more vulnerable to this problem than other languages. FILIGRANE offers a complete framework and the software community addresses aspects such as registration of the mobile code, certification of the entities involved in e-commerce scenarios, copyright protection, protection against modifications and malicious use. We use ERMS (Electronic Right Management System) which manages all kinds of rights and contract handling associated with a piece of mobile code on the provider platform. This paper describes different security blocks and their integration into the FILIGRANE security framework as well as the integrated architecture in the web. We make extensive use of labelling and watermarking technologies to embed information into to the software as well as their associated reverse process tools to extract the label and watermark from the Java byte code. Obfuscation technology is used to make the reverse engineering as difficult as possible. We use standard Java Cryptography Architecture (JCA) and Java Cryptography Extension (JCE) as underlying security infrastructure. We use smart cards as a secure token to save the copyright information and secure communication based on public key methodology. We adapted the background from OKAPI [10] and OCTALIS [2] projects to the FILIGRANE developments.

2. General Architecture

Below we give the list of the actors associated with the FILIGRANE functional model. The model is influenced by the IMPRIMATUR [3] project. The FILIGRANE system is composed with the following actors:

- **Certificate Authority (CA)** : a Trusted Third Party(TTP), providing services for the creation and distribution of electronic certificates for producer, end user, provider and the Rights Clearing House (RCH);
- **Card Issuer**: This is the entity providing the registered end users of the FILGRANE system with personalized smart cards;
- **Producer** : a software developer or company, offering mobile code to a provider for E-commerce;

[1] corresponding author: jalali@igd.fhg.de, Tel:+49-6151-155532, Fax:+49-6151-155499

- **Provider** : actor providing goods such as software, services or information. Provider sells services and/or electronic delivery of items for sale such as software. Provider negotiates the contract conditions for the use of services electronically;
- **End User** : an authorized holder of a certificate supported by a CA and integrated in a personalized smart card by the Card Issuer , and registered to perform software download by the FILIGRANE system;
- **Rights Clearing House** : this actor is an extension of the IMPRIMATUR IPR Database. The FILIGRANE Rights Clearing House is dedicated to the definition and redistribution of rights between actors of the system as the result of a transaction;
- **Fee Collecting Agency** : this actor holds the responsibility to collect funds as a result of financial transactions and to redistribute them proportionally to the various actors of the system according to the conditions of the associated contracts. This operation requires a tight linkage between the Fee Collecting Agency and the Rights Clearing House;
- **Quality Label Service** : this optional actor can enter the system with the role of qualifying a mobile code to be distributed with various quality labels recognized by potential purchasers;
- **E-Notary** : this actor will notarize all transactions in the system and acts as a trusted repository for all actors. The role of the E-Notary is much wider than the role of the Monitoring Service Provider (MSP) in the IMPRIMATUR model. In the case of FILIGRANE the E-Notary notarizes all events (down to individual usage of mobile code by purchasers) in order to constitute a full database guaranteeing the rights of ALL actors in the system.

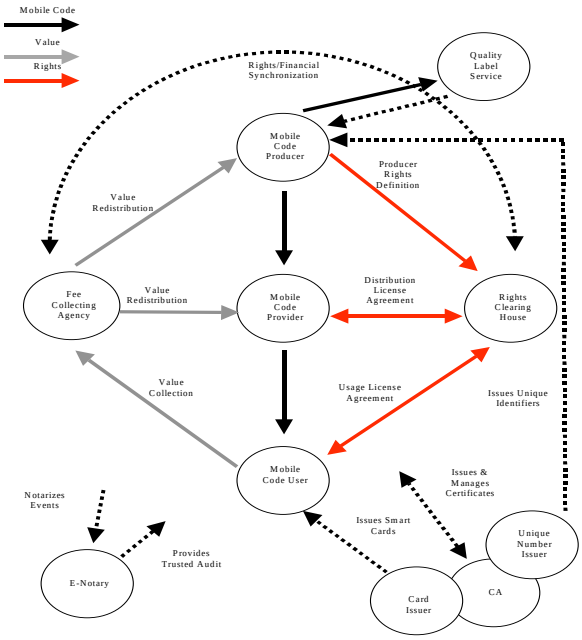

Figure 1 : FILIGRANE Actors

3. Filigrane Security Blocks

By combining of a number of security blocks we provide a secure framework for the commerce of mobile code. These security blocks are presented below.

3.1 Protection of the Mobile Code Container

3.1.1 The Protection of the Code Itself

Different kinds and levels of protection can be achieved with the mobile code. The Filigrane architecture will provide support for some protection mechanisms. Those mechanisms are described in this section.

3.1.1.1 Signature

The mobile code can be signed. This signature can guarantee the origin and the integrity of the mobile code. Note that more than one entity can sign the mobile code. The code producer(s), the code provider(s) can sign the mobile code or a part of it (if the components of the mobile code come from different producers). Each file inside the mobile code can be individually signed. Other security elements can also be signed.

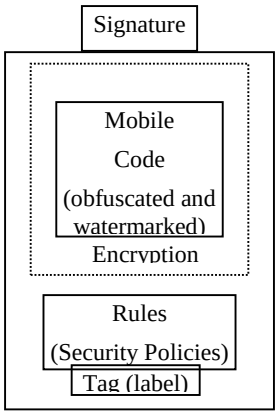

Figure2: FILIGRANE secured container

3.1.1.2 Encryption

The encryption of the mobile code has two objectives. The first objective is to avoid a decompilation / reverse engineering. The second is to control parts of the execution / read rights of the customer. The mobile code is decrypted only if the customer has the rights to execute / read the mobile code. As for the signature, this encryption can be made on individual files inside the mobile code.

3.1.1.3 Rules

Rules can be associated with the mobile code. These rules will describe some parts of the contract between the producer, the provider and the end-user. These rules will be checked by the Filigrane execution environment to avoid any breach of the contract.

3.1.1.4 Watermarks/Cryptomarks

Another stage of protection is the presence of watermarks/cryptomarks embedded within the mobile code. The purpose of these marks can be multiple: identification of the code, integrity check, avoid the decompilation/reverse engineering of the mobile code. An important difference with other protection mechanisms is that these marks are embedded within the code and not associated with it, it implies that the mobile code itself can

verify the presence of these marks. Cryptomarks are verified by the mobile code when it runs on the client system.

3.1.1.5 Obfuscation

Intuitively, the obfuscation of the code has purpose to make modifications in the compiled code in order to make it harder to reverse engineering. This can have two meanings and both are interesting here:

- Alter the code to make it very difficult to decompile. (Target: the decompiler/deobfuscator);
- Alter the code to make it incomprehensible once decompiled. (Target: the engineer).

3.1.1.6 Tagging (Labeling)

A tag is an important data. It permits to identify the mobile code (name, version, author, date ...). That is why a label must be present with the code. Note also that a part of the (or maybe the whole) tag must be signed to protect the data embedded in it.

3.1.2 The Protection of the Code Envelope

All these protection mechanisms must be combined inside a whole package. That is the objective of the code envelope. The code envelope will provide an integrated package where the mobile code and all protections associated with this mobile code are stored.

3.2 ERMS

ERMS stands for Electronic Right Management System and is the main security block to produce and control rules. ERMS checks the rules before and during the execution. The preliminary set of rules supported by FILIGRANE at the present are:

- At runtime

  Time limit: Interval of dates and usage time.

- Before execution

  Versioning

ERMS use a new language called SRDL standing for Software Rights Description Language is based on XML. SRDL is designed for right management in FILIGRANE.

3.3 SmartCard

The FILIGRANE Framework uses crypto engine smart cards as:

- Secure physical token for storage of usage rights
- Secured end links for the PKI enabled trusted environment.

The mobile code can be only downloaded from a user who owns a valid and registered smart card. The smart card must be inserted into the card reader during the download process. The counter and usage keys for the link between the smart card services on the server and the client side is secured using a one way authentication protocol. This is the most important security feature in FILIGRANE. The security checks will be done before the downloading process can begin.

4. Filigrane Security Engine

All operations on the mobile code within the FILIGRANE trusted environment must be controlled by a Security Engine.

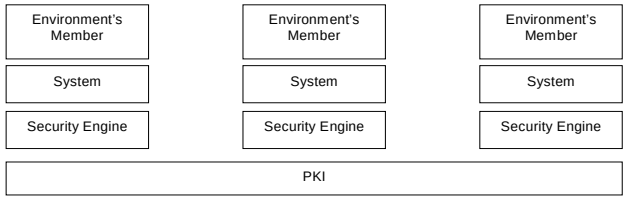

Figure 3: FILIGRANE Functional Diagram

The FILIGRANE security engine must be embedded in the mobile code environment of hosts to control mobile code packaging and

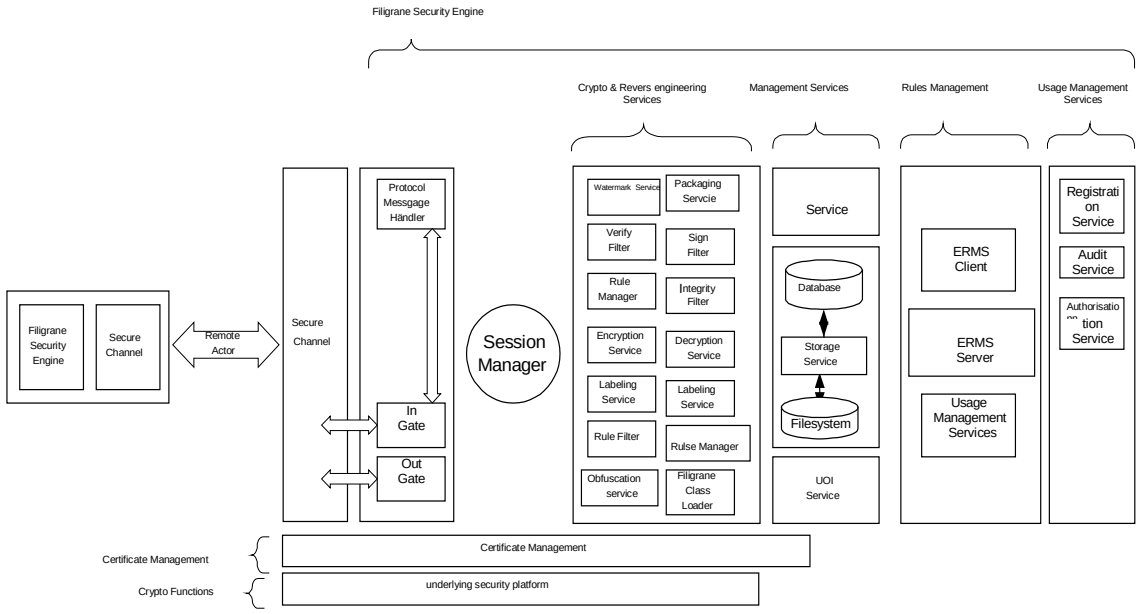

Figure 4: Filigrane Security Engine Functional Blocks

execution. The execution environment could be a browser or a mobile agent platform or every other kind of mobile code system. The following environment is needed for code execution under FILIGRANE:

- A FILIGRANE security engine must be plugged into the Mobile Code Provider platform in order to control transactions (distribution, deposit).
- A FILIGRANE security engine must be used by the Mobile Code Producer to initially prepare and deposit his code for distribution.
- A FILIGRANE security engine must be implemented at the end user execution environment.

Membership to a FILIGRANE Trusted Environment implies:

- Inscription in the Environment's PKI.
- Equipment of systems with the FILIGRANE security engine.

We designed the FILIGRANE security engine to be independent of the communication channel in order to be pluggable in every environment and to be equipped with copyright protection for its mobile software.

The FILIGRANE security engine consists of a number of configurable named services. A *service* consists of instance and name where *instance* is the object implementing the service interface, namely the well-known name of the service. Depending on the entity, particular services could be added to service registry. The relationship between the security engine and the rest of FILIGRANE is controlled via the Session Manager. Using this approach the services in FILIGRANE could be implemented by different people and are easy to plug onto the server. Services are registered and managed by the FILIGRANE *Session Manager*. In practice, special services for every entity could reside in a particular server directory from which they are loaded automatically when the server boots or could be downloaded from other locations in the Internet if they are not available locally. In the case of non- availability of the security blocks the way how these blocks could be downloaded into the FILIGRANE security engine is matter of discussion. Fraunhofer-IGD is investigating solutions based on dynamic secure class loading for needed classes, objects and components over the network at runtime. Our project partner GEMPLUS is investigating the use of a part of the OPIMA (Open Platform Initiative for Multimedia Access) [7] for this task. Figure 4 explains the basic architecture of the FILIGRANE security engine and its functional security blocks.

5. Filigrane Protocol

Interactions between the entities can occur in an interactive environment, such as the World Wide Web, or through non-interactive means such as electronic mail exchanges. The protocol steps could be processed online or offline and for any kind of digital object such as mobile code or digital images, videos, etc. For this reason we defined a detailed protocol based on the OCTALIS [2] model. This is an ESPRIT project in which Fraunhofer-IGD is involved in. The goal of the project was to develop security protocols for trading digital images over the Internet. We will describe the FILIGRANE protocol in an extra paper.

The FILIGRANE protocol will enable different parties which are distributed in a network to understand each other and to put all entities to a functioning distributed system. FILIGRANE does not currently support a payment phase, but we are investigating to adapt existing technologies for this reason.

6. Implementation and Web Integration

All security blocks except the watermarking service are implemented and integrated into the web application. We have integrated the FILIGRANE security framework into the web infrastructure and are planning to make our first trial with real web users and applets according to the illustration in Figure 5 on the next page.

For the integration into the Web, we use existing Internet technologies as well as Java capacities. At the client and the server side we need components to forward protocol messages to the FILIGRANE runtime and we need a secure channel to send the FILIGRANE messages to the receiver. In particular we use following technologies:

HTTP Server: for the provider and **HTTP Client** (Browser) for the end user.

Java PlugIn: Technology on the client side in order to execute signed applets with the new JKD1.2 classes and locally installed FILIGRANE classes. The existing browser implementations support only old JKD versions.

HTTP Servlet API: on the server side in order to filter the FILIGRANE messages at the HTTP server side and forward them to the FILIGRANE run time and present the results back to the client.

SSL (Secure Socket Layer) for: Confidentiality, Integrity Authentication, Non-repudiation of the e-commerce transaction.

RMI (Remote Method Invocation): This is used as an extra channel in order to contact the remote card and save the counter, DESKey and the usage rights into the remote smart card.

Based on this architecture we have realized the following phases of the mobile code commerce in the Web:

Ordering, Browsing, Negotiation, Selection of item, Contract handling, Authorization, Confirmation, Delivery of goods (Mobile Code), Execution in a Controlled Environment.

At the moment the Web application consists of two actors from the listed actors in the general architecture described above:

- A registered user on the client side who is using a browser to download the mobile code
- The provider on the server side who is running an HTTP server and offers the mobile code to registered users.

Our first demonstrator merges the provider and the producer of the mobile code to the same entity. Figure 5 illustrates the overall design of this integrated Web architecture. In this version of the demonstrator we have integrated the ERMS system to the FILIGRANE server. The ERMS consists of different parts for server and the client. ERMS on the server side is the main entry point to the FILIGRANE mobile code server and central entity for right management. ERMS controls all right management questions before the mobile code is transmitted to the client and the right checks before the execution on the client platform. The counter and symmetric key for the usage of the content (mobile

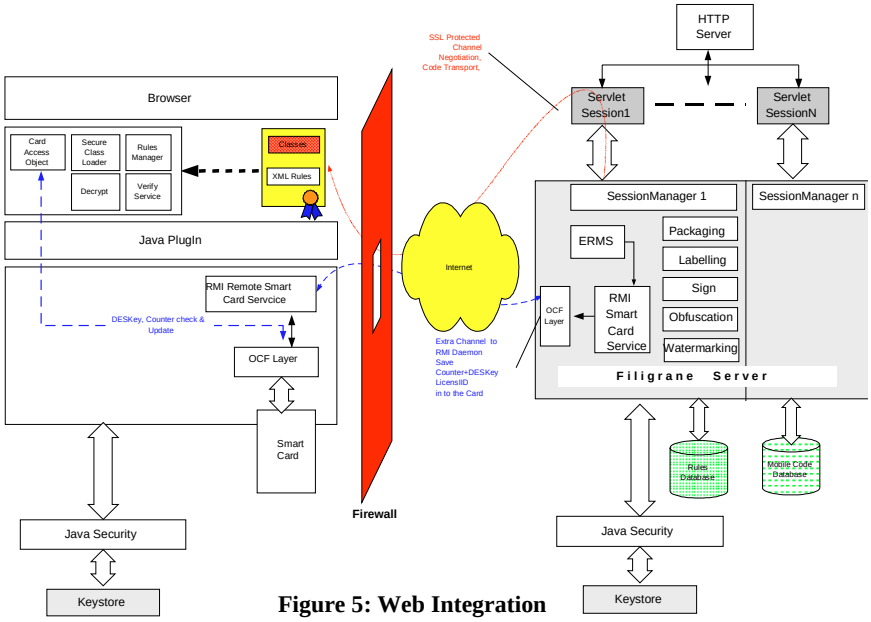

Figure 5: Web Integration

code) are sent to the remote card using the RMI channel. We use the SSL protected channel for confidentiality, authenticity, non repudiation and integrity of the e-commerce transaction. The link between the FILIGRANE server and the remote card is secured using the security functionality of the smart card. The description of the web application as well as the dynamic model of the web scenario ??an key and certificate management will be published in the next future.

7. Related Work

A number of companies, research institutes and organizations [2, 7,...] are working on concepts, architectures and protocols for the copyright protection of multi media data. Some projects and developments are trying to solve this problem for still/video pictures and sounds with success. However, the problem is much more complex when we consider the software. Some other important projects and development are under work and managed by big companies such as IBM. However, these projects are mainly keyed to the securing of Java itself. FILIGRANE as described above addresses all security aspects of dynamic distribution of the mobile code from production to usage and the monitoring of mobile Java software. Software & Information Industry Association [1] try to put the code and content industries, business, development, research and education into a global consortium.

8. Conclusions

Internet software piracy is growing rapidly as network use spreads and could surpass traditional forms of piracy. There are a number of good security blocks to protect the mobile code itself, but most of the solutions are loose and not integrated into a global framework. FILGRANE is the first attempt to provide the web community with a global framework with all aspects of software protection. We know that FILIGRANE solutions will not enable the developers to protect their Software with absolute guaranty, but we will make the software piracy much more difficult than it is now. The result of our work will be tested using two different agent platforms. The first one is a FIPA compliant mobile agent platform called Jade [8] developed by CSELT in Italy, the second one is a security centric mobile agent platform called SeMoA (Secure Mobile Agents) [4] developed by Fraunhofer-IGD icd Germany. The basic security concepts of SeMoA are published mainly in [5] and [6]. The actors listed in the functional model are not implemented completely but we are working continuously on it.

9. ACKNOWLEDGMENTS

Parts of this work were sponsored through the ESPRIT project FILIGRANE (FlexIbLe IPR for Software AGent ReliANcE), project number 28423. We would like to thank our partners, in particular Dr. Rigobert Foka, Dr. Catherine Simon, Michel Reguidel from Thomson-CSF in France for their valuable project management efforts.

10. REFERENCES

[1] Software & Information Industry Association (SIIA) http://www.siia.net/

[2] OCTALIS (Offer of Contents through Trusted Access LinkS), http://www.igd.fhg.de/igd-a8/projects/octalis/octalis_en.html

[3] Imprimatur ESPRIT Project, Project Number 20676, http://www.imprimatur.alcs.co.uk/

[4] Secure Mobile Agents Project, http://www.igd.fhg.de/igda8/projects/semoa/semoa_de.html

[5] Concepts and Architecture of a Security-centric Mobile Agent Server, Volker Roth and Mehrdad Jalali, submitted to ASA/MA'99 Third International Symposium on Mobile Agents

[6] Access Control and Key Management for Mobile Agents, Volker Roth and Mehrdad Jalali, Computer & Graphics, Vol. 22, No. 4, pp. 457-461,1998

[7] OPIMA Open Platform Initiative for Multimedia Access http://drogo.cselt.it/ufv/leonardo/opima/

[8] Java Agent Development Framework, http://sharon.cselt.it/projects/jade/

[9] Open Kernel for Access to Protected Interoperable interactive services http://www.tele.ucl.ac.be/OKAPI/index.html

[10] Open Kernel for Access to Protected Interoperable interactive services http://www.tele.ucl.ac.be/OKAPI/index.html